Part 1. Implement the classes Animal and Zoo as presented in the following UML diagram

Zoo
- animals: Animal[]
+Zoo(setOfAnimals : String) +getAnimals(): Animal[] +countCats(): int +countYoung(): int +toString(): String

Animal
- type : String - color : String - age : int
+Animal (type: String, color: String, age: int) + getters + isYoung(): boolean
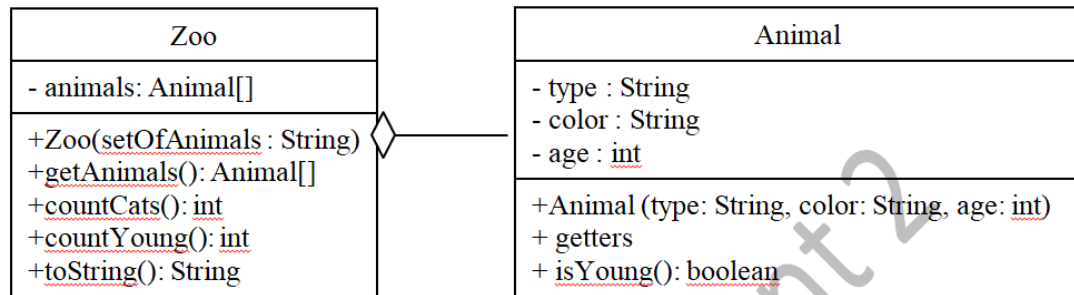
The class Animal contains:
- Fields:
    - The animal's type: must be cat or bird.
    - The animal's color.
    - The animal's age in months.
- Methods:
    - All-arguments constructor.
    - All getters

The class Zoo contains:
- Fields:
    - An array of animals.
- Methods:
    - A constructor that creates a Zoo abject. The constructor takes as parameter a String formed of a set of animals' descriptions separated by a dash (-), as in the following example:
      black cat 2 – red bird 6 – white cat 12 - ....
      The constructor extracts the information about each animal (type, color, age) and fill the array animals.
      We assume that each animal's description follows the format:
      color[space]type[space]age (example black cat 2)
    - The getter of the array animals
    - A method countCats that returns the number of cats in the zoo.

o A method countYoung that returns the number of young animals in the zoo.

o A method toString that prints the information about the Zoo as in the following example:

The Zoo contains 4 cats and 2 birds.

Part 2. Implement a driver class to test the classes in Part 1 as follows:

a. Read from the user a String formed of animals' descriptions and create a Zoo object
b. Display the count of young animals in the Zoo.
c. Display the information about the Zoo.
d. Without adding extra methods to the classes Zoo and Animal, write in the driver the code to search for and display the number of red animals in the Zoo.

Sample run

```
enter a set of animals
black cat 2-red bird 6-white cat 12
2
The Zoo contains 2 cats and 1 birds
the number of red animals in the zoo is 1
```